

Nattan MAISONNEUVE–FLEURY
BTS SIO 1
Groupe B

20/11/2025

Compte Rendu - Scene Builder

Sommaire :

Introduction :	2
2. Configuration de l'Environnement	2
3. Création du Projet	2
4. Conception de l'Interface avec Scene Builder	3
5. Implémentation du Contrôleur (MainController.java)	4
6. Implémentation de l'Application (MainApp.java)	4
7. Test et Validation	4
Conclusion :	5

Introduction :

Dans ce TP, nous allons voir comment configurer un environnement de travail avec JavaFX.

2. Configuration de l'Environnement

Pour réaliser ce projet, les étapes de configuration suivantes sont nécessaires :






- **Logiciel de Développement** : Utiliser l'environnement de développement **BlueJ** (disponible sur www.bluej.org/).
- **Installation de JavaFX** : Télécharger et installer le **SDK JavaFX** depuis le site de Gluon (<https://gluonhq.com/products/javafx/>).
- **Configuration de BlueJ** :
 - Dézipper le fichier téléchargé. Le dossier extrait contient les sous-dossiers **bin**, **legal**, et **lib**, ainsi qu'un fichier **src.zip**.
 - Ouvrir BlueJ, aller dans **Outils > Préférences**.
 - Dans l'onglet Bibliothèque, ajouter tous les fichiers **.jar** contenus dans le dossier **lib** en cliquant sur le bouton **Add**.
 - Redémarrer BlueJ pour finaliser la configuration.

3. Création du Projet

1. **Créer un nouveau projet** dans BlueJ, nommé par exemple *Scenebuilder*.
2. **Créer les classes Java** : Deux classes sont nécessaires, *Main* et *MainController*.

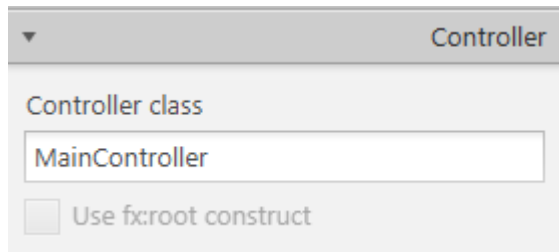


3. **Créer le fichier FXML** : Étant donné que BlueJ ne le génère pas automatiquement, créer manuellement un fichier nommé *Mainview.fxml* dans le répertoire du projet : `C:\Users\leleve\Scenebuilder`

 Main.java	09/12/2025 14:11	Fichier JAVA	1 Ko
 MainController.java	09/12/2025 14:11	Fichier JAVA	1 Ko
 package.bluej	09/12/2025 14:13	BlueJ Project F...	1 Ko
 README.TXT	09/12/2025 14:11	Document texte	1 Ko
 Mainvieaw.fxml	09/12/2025 14:20	Fichier FXML	0 Ko


4. Conception de l'Interface avec Scene Builder

1. **Ouvrir Scene Builder** séparément.
2. **Charger le FXML** : Cliquer sur *Open project* et sélectionner le fichier *Mainview.fxml* créé précédemment.
3. **Définir le Contrôleur** : Dans la section **Controller class**, définir la classe de contrôle comme *MainController*.





4. Ajouter les Composants de Base :

- Rechercher *AnchorPane* dans la Library (colonne de gauche) et le glisser-déposer sur la zone de travail (colonne du milieu).

 AnchorPane

- Ajouter un TextField (pour le nom d'utilisateur) et un PasswordField (pour le mot de passe).


 TextField

 PasswordField

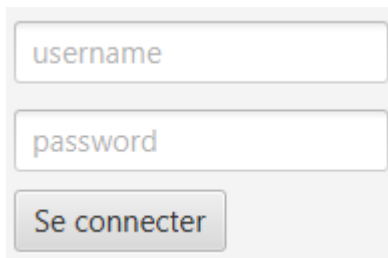
- En même temps sur la droite ajouter un nom pour informer l'utilisateur

Prompt Text

- Ajouter un Button (texte : *Se connecter*).

 Button Se connecter

- Voici le résultat :



5. Ajouter les Messages de Statut :

- Ajouter deux Labels supplémentaires, un pour le message d'erreur

Identifiant ou mdp incorrect

Visible ☐

et un pour le message de réussite :

Connexion réussie

Visible ☐

6. **Définir les Identifiants (fx:id)** : Pour que Java puisse reconnaître les éléments, définir leurs fx:id dans la section Code de la colonne de droite.
 - Champ Utilisateur : *usernameField*.
 - Champ Mot de passe : *passwordField*.
 - Label d'erreur : *errorLabel*.
 - Label de succès : *successLabel*.
7. **Définir l'Action du Bouton** : Pour le bouton Se connecter, ajouter la méthode *#handleLogin* dans la section On Action.

5. Implémentation du Contrôleur (MainController.java)

1. **Récupérer le Squelette** : Dans Scene Builder, aller dans View puis cliquer sur Show Sample Controller Skeleton pour générer le code de base du contrôleur.
2. Copier le code et le coller dans la classe MainController.java dans BlueJ.
3. Remplir la méthode handleLogin pour gérer la logique de connexion (vérification des identifiants) et l'affichage des labels de statut.

Les identifiants de test utilisés sont *admin* pour l'utilisateur et *1234* pour le mot de passe.

Le code complété doit vérifier les identifiants et basculer la visibilité des labels:

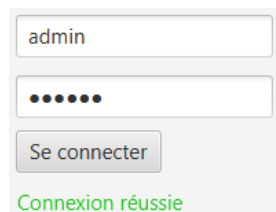
- Si succès : *successLabel* visible, *errorLabel* invisible.
- Si échec : *errorLabel* visible, *successLabel* invisible.

6. Implémentation de l'Application (MainApp.java)

Remplir la classe MainApp pour lancer l'application, en chargeant la vue FXML (Mainview.fxml).

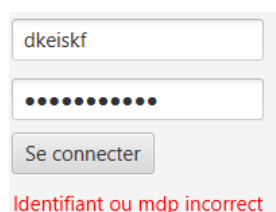
7. Test et Validation

1. Faire un clic droit sur la classe MainApp dans BlueJ et sélectionner l'option **Run JavaFX Application**.
2. L'application de connexion s'affiche.
3. Tester la connexion avec les identifiants corrects (*admin/1234*) pour obtenir le message **Connexion réussie**.



The screenshot shows a login form with two input fields. The first field contains the text 'admin'. The second field contains six dots, representing a masked password. Below the fields is a button labeled 'Se connecter'. At the bottom of the form, the text 'Connexion réussie' is displayed in green.

4. Tester avec des identifiants incorrects pour afficher le message **identifiant ou mdp incorrect**.



The screenshot shows the same login form as before. The first input field now contains 'dkeiskf' and the second field contains ten dots. The 'Se connecter' button is still present. At the bottom, the text 'Identifiant ou mdp incorrect' is displayed in red.

Conclusion :

Ce TP a permis de maîtriser les bases de la création d'une application de bureau avec JavaFX.

Nous avons réussi à :

1. Configurer l'environnement (JavaFX, BlueJ).
2. Concevoir l'interface de connexion graphiquement avec Scene Builder (FXML).
3. Implémenter la logique de vérification des identifiants dans le contrôleur Java.

L'exercice a démontré l'efficacité de la collaboration entre le FXML (Vue) et le Contrôleur Java, posant les fondations pour le développement d'applications JavaFX plus complexes en utilisant le modèle MVC.